

TP2 - Codage et compression multimédia

Changement d'espace couleur

Léa Serrano M1 IMAGINE

Lien de mon git pour ce tp : <https://github.com/LeaSerrano/M1-IMAGINE-CompressionDonnees-TP2.git>

Table des matières

1	Ex 1	2
2	Ex 2	5
3	Ex 3	7

1 Ex 1

(questions 1 à 3 du tp)

Le but de cet exercice est de prendre deux des composantes RGB d'une image couleur et de réduire leur taille pour avoir une image deux fois plus petite. Nous réaliserons cela en moyennant 4 pixels et en mettant le pixel de la moyenne dans une nouvelle image plus petite. Cela va nous permettre de diviser par 2 la taille de l'image (au lieu d'avoir une image en 512x512, nous aurons une image en 256x256). Ensuite nous allons ré-échantillonner (en utilisant une interpolation bilinéaire) ces deux composantes réduites pour qu'elles retrouvent leur taille initiale.

Puis nous allons les rassembler avec la dernière composante (qui n'aura pas été ré-échantillonnée) pour retrouver une image RGB.




Enfin, nous allons regarder le PSNR entre l'image de base et l'image obtenue afin de voir si l'image a été altérée par la compression.

Voici l'image que nous allons utiliser :



Voici chaque composante R, G et B de cette image dans la même taille que l'image de base :

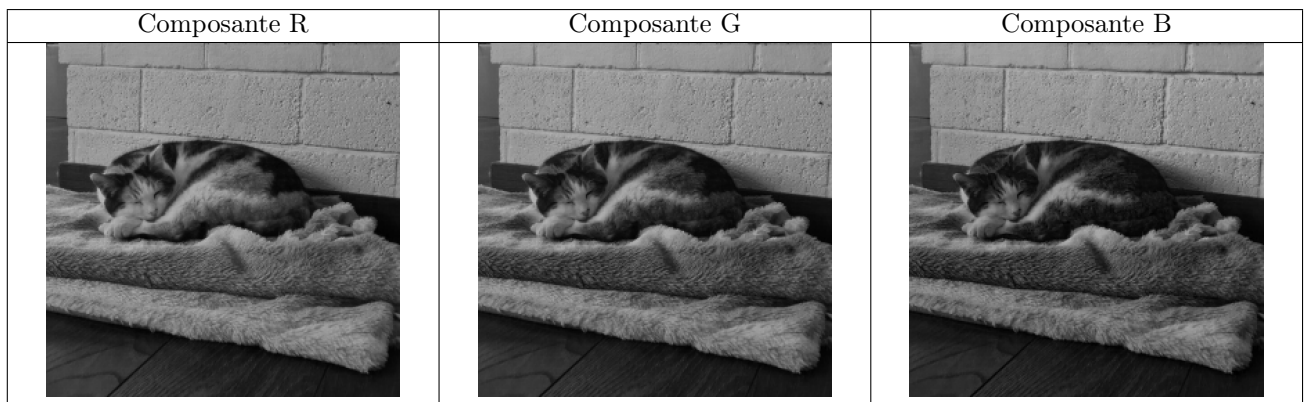
J'ai dû réduire la taille des images pour le CR car les images étaient trop grandes, donc les images du CR ont une taille représentative (mais lorsque je dis que la taille est la même ou a été réduite, c'est vraiment le cas, cela peut-être vérifié sur mon git dans le dossier : ImagesTP).

Composante R	Composante G	Composante B
		





Ensuite on va réduire la taille de chaque composante R, G et B :



Ensuite on va ré-échantillonner nos images pour qu'elles retrouvent leur taille initiale :



Enfin on va prendre une image R, G ou B non transformée et deux images R, G, ou B ré-échantillonnées et les assembler pour avoir une image RGB :

Image RGB de base	Composante R non transformée, G ré-échantillonnée et B ré-échantillonnée
	
Composante R ré-échantillonnée, G non transformée et B ré-échantillonnée	Composante R ré-échantillonnée, G ré-échantillonnée et B non-transformée
	

On voit que les 4 images sont quasiment identiques.

On va ensuite calculer le PSNR entre chaque nouvelle image et l'image de base :

```
Donnees-TP2$ ./PSNR chat.ppm recomposition_cR_eG_eB.ppm
PSNR : 32.3989
lea@lea-Swift-SF314-59:~/M1-IMAGINE/S2/CompressionDonnées/M1-I
Donnees-TP2$ ./PSNR chat.ppm recomposition_eR_cG_eB.ppm
PSNR : 32.3937
lea@lea-Swift-SF314-59:~/M1-IMAGINE/S2/CompressionDonnées/M1-I
Donnees-TP2$ ./PSNR chat.ppm recomposition_eR_eG_cB.ppm
PSNR : 32.3915
```

Chaque image a quasiment le même PSNR.

2 Ex 2

(question 4 du tp)




Maintenant nous allons réaliser quasiment la même chose que précédemment mais cette fois nous allons traiter des images de l'espace YCrCb.

Nous allons donc appliquer les mêmes traitements aux images Cr et Cb de notre image (diminution de la taille puis ré-échantillonnage).



Puis rassembler l'image Y et les images Cr et Cb ré-échantillonnées puis former une image RGB.

Enfin, nous allons comparer cette image obtenue avec l'image de base en utilisant le PSNR.

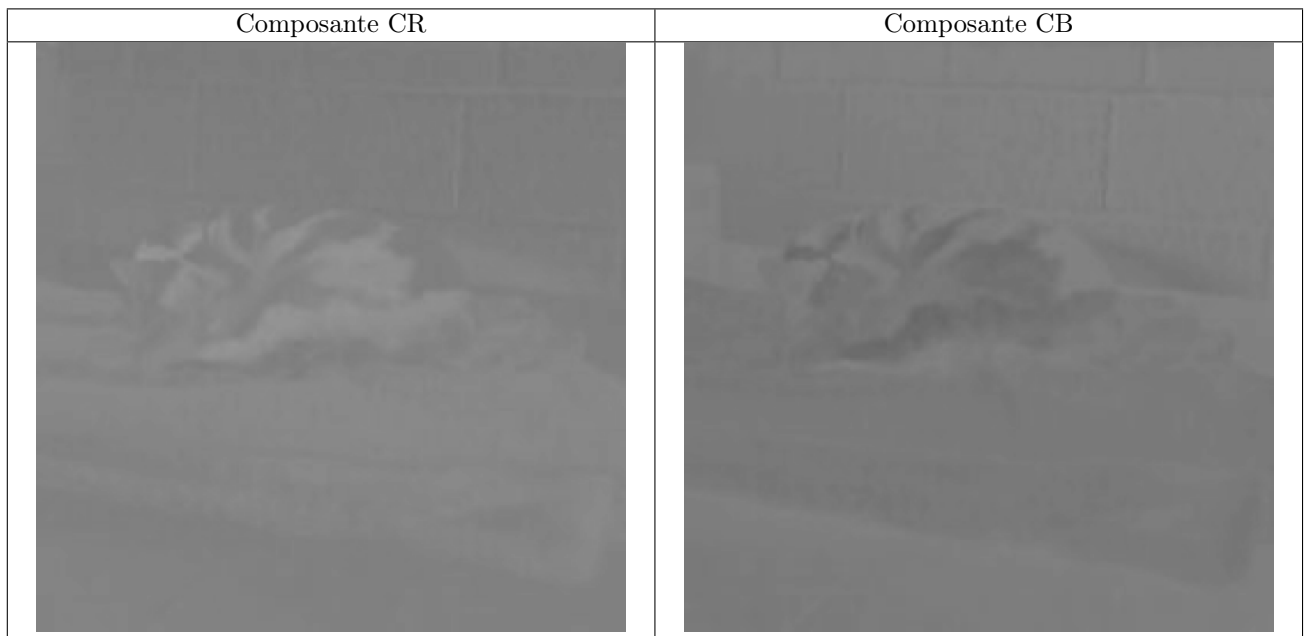
Voici les composantes Y, Cr et Cb de notre image :

Composante Y	Composante CR	Composante CB
		

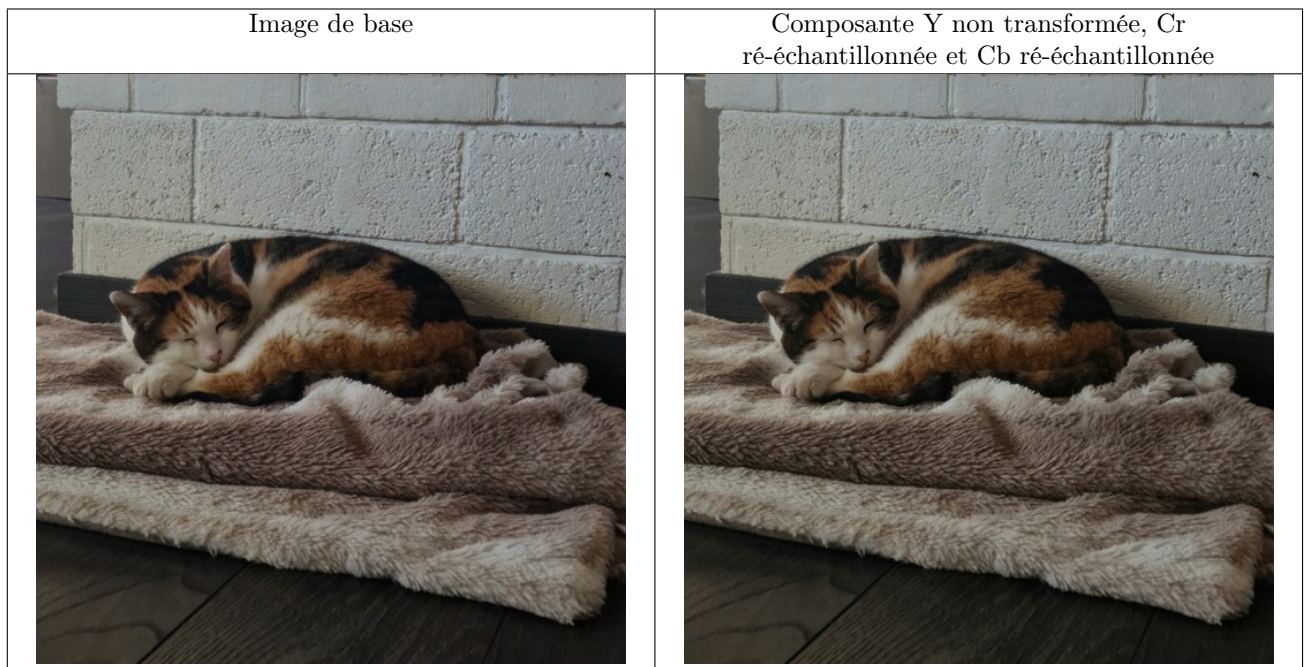
On va ensuite réduire la taille de nos images Cr et Cb :

Composante CR	Composante CB
	

Puis nous allons ré-échantillonner nos deux images :



Enfin nous allons combiner ces deux images avec celle du Y qu'on avait obtenu au début pour obtenir une image en couleur :



Pour comparer leur ressemblance , on va calculer le PSNR entre les deux images :

```
Donnees-TP2$ ./PSNR chat.ppm reconstructionYCrCb.ppm
PSNR : 42.8084
```

3 Ex 3

(question 5 du tp)

Le PSNR est un indicateur de la qualité d'une image. Plus le PSNR entre une image compressée et son image originale est élevé, moins l'image compressée a été altérée.

Ici on voit que dans l'espace RGB, on a un PSNR qui est aux alentours de 32.39 peu importe les composantes ré-échantillonnées.

Dans l'espace YCrCb, on a un PSNR qui est à environ 42.8.

La compression dans l'espace YCrCb ayant un meilleur PSNR que celle dans l'espace RGB, on peut en conclure que la meilleure compression est celle dans l'espace YCrCb.

D'autres méthodes existent pour avoir un taux de compression de 2, au lieu de faire la moyenne on pourrait : sous-échantillonner par facteur de réduction (on pourrait ne prendre qu'un pixel sur 2) ou encore sous-échantillonner par filtrage (utilisation d'un filtre pour réduire le nombre de pixels).